CONV-TTL-BLO
Multiboot HDL module

Carlos Gil Soriano
BE-CO-HT
carlos.gil.soriano@cern.ch

February 23, 2012

Abstract

The *multiboot module* is in charge of configuring multibooting capability and assert the reprogramming of the FPGA.

This document shows:

- Parameters used as *generic*
- The registers to control the module.
- Step-by-step instructions for proper use.

Revision history		
HDL version	**Module**	**Date**
0.1	Multiboot manager	February 23, 2012

Contents

1 Structure

NOTE1: this module is platform specific. It only works with Spartan 6
NOTE2: in case the EEPROM memory is replaced, SPI opcode will change. User should notice this issue.

The trigger module contains sever blocks related the following way:

– ***multiboot_top.vhd***
—– **multiboot_regs.vhd**
—– **multiboot_core.vhd**
——— ICAP_SPARTAN6 (*Xilinx primitive*)

1.1 *multiboot_top.vhd*

The top file of the module. It interconnects the Wishbone to internal register module, *multiboot_regs.vhd*, to the core logic in *multiboot_core.vhd.*

No *generics* are implemented in this HDL module.

1.2 multiboot_regs.vhd

In this module the registers neeeded for specifiying the memory addresses in which the FPGA must boot to are defined.

An internal register is defined for selectively controlling operations to be performed by this module (full ICAP reprogramming process, issuing ICAP commands, refreshing ICAP registers). The set of operations that can be issued is restricted for security reasons. The allowed operations are further listed in the *Register subsection.*

1.3 multiboot_core.vhd

It is responsible of accessing ICAP port through the internal *ICAP_SPARTAN6 Xilinx primitive.* A finite state machine is implemented in accordance to Chapter 7 of [1].

1.4 Behaviour

Following the instructions of [1] strictly leads to correct multiboot of the FPGA. Firstly, registers *GENERAL1*, *GENERAL2*, *GENERAL3* and *GENERAL4* must be programmed with valid values. It should be keept in mind that the *SPI opcode* in *GENERAL4* register depends on the *EEPROM chip* mounted on the board.
Then, a *full multiboot* command must be performed via ICAP interface through a write in *CTRL* register in *multiboot module.*

2 Parameters

No *generic* parameters are offered in this module.

3 Registers

3.1 CTRL

The *CTRL* register is a read-write register for *OP* field and a read-only for *PEND* bit. It specifies the operations that can be controlled by an user.

Bits	**Field**	**Meaning**
3-0	OP	OPeration to be performed
4	PEND	operation PENDing

Whenever an operation is specified by the user, it is passed to ICAP Xilinx primitive through *multiboot_core.vhd* and the bit flag *PEND* is set to '1' until it is completely finished.

3.1.1 Operations

The valid operations that can be requested are the following:

OP byte	**Operation**
0x0	**Full multiboot process** as specified in [1]
0x1	**Write GENERAL1** register from *multiboot_regs.vhd* into FPGA
0x2	**Write GENERAL2** register from *multiboot_regs.vhd* into FPGA
0x3	**Write GENERAL3** register from *multiboot_regs.vhd* into FPGA
0x4	**Write GENERAL4** register from *multiboot_regs.vhd* into FPGA
0x7	Perform **IPROG command**
0xD	**Refresh STAT** register into *multiboot_regs.vhd*

Full multiboot process, $OP = $ 0x0, comprises commamnds:

1. $OP = $ 0x1
2. $OP = $ 0x2
3. $OP = $ 0x3
4. $OP = $ 0x4
5. $OP = $ 0x7

3.2 STAT

The *STAT* register is a read-only register. A *refresh operation* should be completed before retrieving correct *STAT* information.

Bits	**Field**	**Meaning**
0	CRC_ERROR	CRC ERROR detected in bitstream
1	ID_ERROR	IDCODE not validated
2	DCM_LOCK	DCMs and PLL are locked
3	GTS_CFG_B	Global tristate
4	GWE	Global Write Enable
5	GHIGH_B	GHIGH
6	DEC_ERROR	DEC_ERROR
7	PART_SECURED	Decryption is set
8	HSWAPEN	HWSAPEN
11-9	MODE	MODE pins
12	INIT_B	INIT_B
13	DONE	DONE input pins
14	IN_PWRDWN	suspend status
15	SWWD_STRIKEOUT	config error because of invalid sync

3.3 GENERAL1

Bit scrambling is done in VHDL code. Bit order must be as specified below:

Bits	**Field**	**Meaning**
15-0	MBT_ADDR_L	MultiBoot image ADDRess Lower half

3.4 GENERAL2

Bit scrambling is done in VHDL code. Bit order must be as specified below:

Bits	**Field**	**Meaning**
7-0	MBT_ADDR_L	Multiboot image ADDRess Lower Half
15-8	SPIO	SPI Opcode

3.5 GENERAL3

Bit scrambling is done in VHDL code. Bit order must be as specified below:

Bits	**Field**	**Meaning**
15-0	GLD_ADDR_L	GolDen image ADDRess Lower half

3.6 GENERAL4

Bit scrambling is done in VHDL code. Bit order must be as specified below:

Bits	**Field**	**Meaning**
7-0	GLD_ADDR_H	GoLDen image ADDRess Higher Half
15-8	SPIO	SPI Opcode

4 Internal Memory Mapping

The Internal Memory Mapping is as follows:

Address	**Register**	**Access**
0x0	*CTRL*	Read-only
0x1	*STAT*	See *STAT* description
0x2	Not used	
0x3	Not used	
0x4	*GENERAL1*	Read-write
0x5	*GENERAL2*	Read-write
0x6	*GENERAL3*	Read-only
0x7	*GENERAL4*	Read-write

5 How to use it

It requieres three parameters to be specified:

- Address of the Golden Image
- Address of the Multiboot Image
- SPI Opcode of the EEPROM serial interface

Bad specifications of addresses will not reprogram the FPGA.

5.1 Submitting ICAP instructions

It can be either a two-step or a single-step process. Two-step processes are related with changes in *GENERAL[X]* register. Submitting an ICAP command is a single-step-process (*IPROG* instruction, for instance).

Example A: Full Multiboot Configuration
This is a scenario is useful when the EEPROM memory map has changed for the allocation of the two FPGA bitstreams.

1. Write *GENERAL1* register.

2. Write *GENERAL2* register.
3. Write *GENERAL3* register.
4. Write *GENERAL4* register.
5. Write *CTRL* register.
   CTRL should issue a **Full multiboot process** operation code (0x0).

Example B: Change an individual Boot Look Up Address
This is a scenario is useful when the EEPROM memory map has changed for the allocation of only one of the FPGA bitstreams.

1. Write *GENERAL[X]* register. Where X=1,3
2. Write *GENERAL[X+1]* register.
3. Write *CTRL* register.
   CTRL should issue a **Write GENERAL[X]** operation code.
4. Write *CTRL* register.
   CTRL should issue a **Write GENERAL[X+1]** operation code.

Example C: Reprogram FPGA without change in Bitstream Location
If the EEPROM memory map has not changed but we want to reload one of the images, we just issue an *IPROG* instruction through the *ICAP* interface.

1. Write *CTRL* register.
   CTRL should issue an **IPROG** operation code(0x7).

References

[1] Spartan-6 FPGA Configuration User Guide. Technical Report UG380 v2.3, Xilinx Inc., 2011. http://www.xilinx.com/support/documentation/user_guides/ug380.pdf.